Lab #3:
Astrometry from CCD Images

James. R. Graham

Your report is due on November 11, 2014 at 6:00 PM PST

1 Overview

In this lab we will use CCD images to measure the positions of astronomical objects relative to the celestial coordinate system. We will use data from the 40-inch (1-m) Nickel telescope at Lick Observatory.

The purpose of this activity is to learn astronomical skills needed to track the position of a minor planet (an asteroid), and thereby determine its parallax and hence the distance. For this lab we will collect data during the first week so that you can get off to a quick start. We will continue to collect data through the end of the month so that you can follow the motion of your target and infer its orbital properties.

This document provides a step-by-step description of raw astronomical CCD images, CCD calibration and correction, measurement of star positions and comparison to standard star catalogs, and measurement of the celestial coordinates of any arbitrary point within the image.

1.1 Key steps

The key steps are:

1) Read FITS files from the CCD on the 40-inch telescope and display the data using the Python astropy.io.pyfits library (§3.1);
2) Apply systematic corrections including bias and flat field (§3.3);
3) Compare your asteroid field image with the digital sky survey and confirm that field is correct (§4);
4) Measure the positions of the stars in your CCD image (§4.1);
5) Cross-correlate the stars in your image with those in the USNO B-1 star catalog (§4.2);
6) Compute the "standard coordinates" for the USNO B-1 stars and match the two lists (§4.3);
7) Perform a linear least squares fit to find the "plate constants"(§5);
8) Examine the residuals between the measurements and the fit;
9) Locate your asteroid in the CCD images;
10) Compute the position of the asteroid and estimate the associated errors;
11) Analyze second and third epoch observations of the same asteroid and estimate the distance to your target (see the handout on measuring parallax.)

1.2 Schedule

This is a four-week lab, with show & tell on 10/21, 10/28, and 11/4. Your lab report is due on 11/11. For "show-and-tell" on 10/21 you should have progressed at least to step 4, and on 10/28 you should be prepared to discuss the residuals in your least squares fit and the error (step 8).

During the last week you should also be working in analyzing the subsequent images of your target (step 11).

2 The Observatory

We will be using the Nickel Telescope[1] for this lab. The Nickel telescope is a general-purpose, 40-inch (1-m) telescope used for astronomical imaging. It is housed in a dome at the north end of the historical Lick Observatory building located on Mt. Hamilton[2] site near San Jose, CA. The Nickel telescope can be used either by local observers or in a real-time, remote control observing mode.

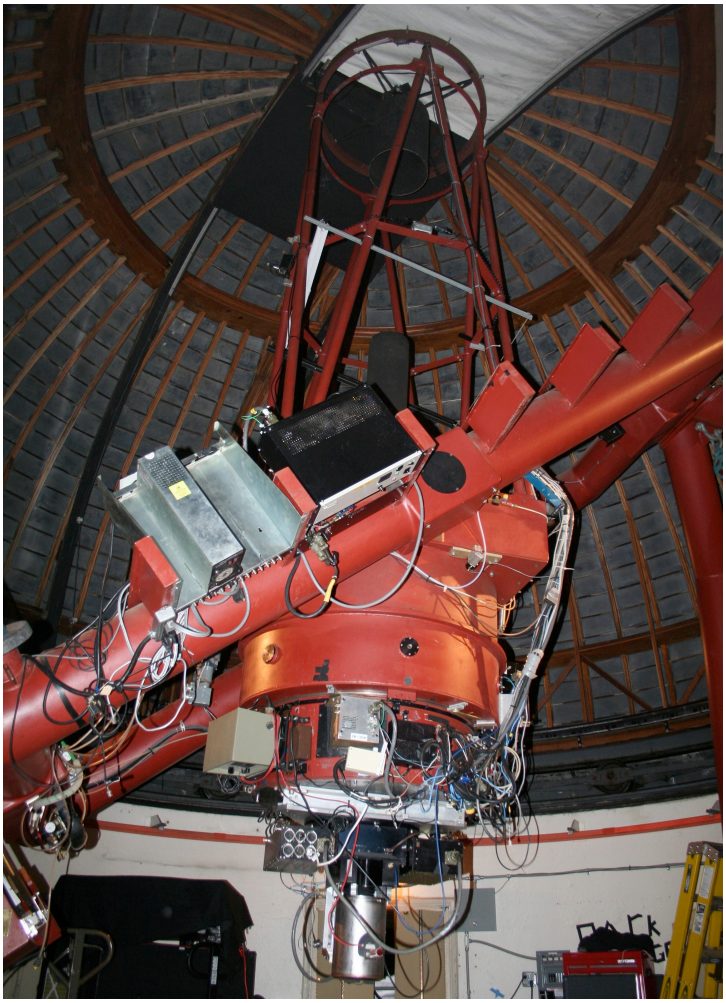

Figure 1: A picture from inside the dome of the 40-inch telescope on Mt. Hamilton. (Photo Credit: Bill Keel, U. Alabama)

The telescope is equipped with a 2048 × 2048 pixel CCD array. The resultant field of view is approximately 6.3 arc minutes × 6.3 arc minutes[1]. Details of the facility are listed in Table 1.

[1] http://www.ucolick.org/public/telescopes/nickel.html
[2] http://www.ucolick.org

Table 1: Nominal observatory and instrument properties

Site	Mt. Hamilton, CA
Geographic location	37°20'49"N 121°37'48"W
Primary mirror	1-m (diameter)
F/ratio-focal length	*F*/17 – 16,840 mm
Camera	CCD-2 (aka Dewar#2)[3]
Detector	LN_2 cooled Loral CCD
CCD format	2048×2048, 15.0 μm pixels
Nominal pixel scale	0.184 arc seconds/pixel (unbinned)
Field-of-view	6.3 arc minutes (square)
Optical Filters	*UBVRI*
Limiting Magnitude	$R \approx 17.5/18.5$ mag. (10/60 seconds)

3 Images

CCD images are saved in a standard astronomical format known as FITS files. FITS files contain the image data, stored in binary format, together with some information that records the circumstances under which the data were obtained.

3.1 FITS headers

Although the image data in FITS files are in binary format, each file also contains an ASCII preamble that can be inspected at the Linux command `fold` by typing, for example:

```
data% fold d159.fits | more

SIMPLE  =                    T / NORMAL FITS IMAGE
BITPIX  =                   16 / DATA PRECISION
NAXIS   =                    2 / NUMBER OF IMAGE DIMENSIONS
NAXIS1  =                 1056 / NUMBER OF COLUMNS
NAXIS2  =                 1024 / NUMBER OF ROWS
CRVAL1U =                 2046 / COLUMN ORIGIN
CRVAL2U =                 2046 / ROW ORIGIN
CDELT1U =                   -2 / COLUMN CHANGE PER PIXEL
CDELT2U =                   -2 / ROW CHANGE PER PIXEL
OBSNUM  =                  159 / OBSERVATION NUMBER
IDNUM   =                    4 / IMAGE ID
OBSTYPE = 'OBJECT'             / IMAGE TYPE
EXPTIME =            20.000000 / Exp time (not counting shutter error)
BSCALE  =             1.000000 / DATA SCALE FACTOR
BZERO   =         32768.000000 / DATA ZERO POINT
COMMENT    Real Value = FITS*BSCALE+BZERO
PROGRAM = 'NEWCAM'             / New Lick Camera
VERSION = 'nickel_direct'      / Data acquisition version
TSEC    =           1374225476 / CLOCK TICK – SECONDS
TUSEC   =                71651 / CLOCK TICK – MICROSECONDS
DATE    = '2013-07-19T09:17:56.07' / UT of CCD readout & descramble
DATASEC = '[1:1024,1:1024]'        / IRAF/NOAO-style data section
COMMENT End of cards hard-coded in fits_cards
COMMENT Begin of cards from other times
ERPBIN  =                    5 / PARALLEL BINNING DURING ERASE
AIRMASS =             1.904125 / AIRMASS AT START OF OBSERVATION
```

[3] http://mthamilton.ucolick.org/techdocs/instruments/nickel_direct/detector/

```
EQUINOXU=          2013.550049 / EPOCH FOR POCO POSITION IS CURRENT DATE
OBSERVER= 'Minkyu Kim, Hae Jung Kim' / OBSERVER NAME
APERNAM = 'Open              ' / APERTURE POSITION NAME
INSTRUME= 'Nickel Direct Camera'
TUB     =             0.000000 / TELESCOPE TUB ROTATION
APERRAW =                 1250 / APERTURE RAW POSITION
FILTRAW =                 2373 / FILTER RAW POSITION
FILTORD =                    2 / FILTER ORDINAL POSITION
APERORD =                    0 / APERTURE ORDINAL POSITION
FILTNAM = 'V                 ' / FILTER POSITION NAME
DATE-END= '2013-07-19T09:17:55.45' / END OF OBSERVATION
DATE-STA= '2013-07-19T09:17:35.45' / START OF OBSERVATION
GEOMCODE=                    0 / READOUT GEOMETRY
DSENSOR = 'Loral 2Kx2K       ' / SENSOR DESCRIPTION
DNAXIS2 =                 2048 / ROWS IN SENSOR
DNAXIS1 =                 2048 / COLUMNS IN SENSOR
UCAMADC = 'Old ADC           ' / UCAM ADC BOARDS
UCAMCDB = 'Old CDB           ' / UCAM CDB BOARDS
CAMERAID=                    2 / CAMERA ID NUMBER
UCAMSPB = '2 DSP             ' / UCAM SPB BOARDS
UCAMSOFT= '4.08 052011       ' / UCAM SOFTWARE VERSION
UCAMTIM = 'Old Timing        ' / UCAM TIMING BOARDS
DEC     = ' 00:05:43.0       ' / DECLINATION
GAIN    =                    1 / DCS GAIN INDEX
TEMPCON =            28.799999 / CONTROLLER TEMPERATURE
COVER   =                   32 / NUMBER OF OVERSCAN COLUMNS
MPP     =                    1 / MPP STATE
HA      = ' 03:15:22.5       ' / HOUR ANGLE
TEMPDET =          -114.400002 / EXPOSURE START DETECTOR TEMPERATURE
TEMPDETE=             0.000000 / EXPOSURE END DETECTOR TEMPERATURE
RA      = '17:44:50.4        ' / RIGHT ASCENSION
OBJECT  = '109_954           '
COMMENT End of cards from other times
DATE-OBS= '2013-07-19T09:17:56.07' /
END
```

Here I have omitted some keywords that encode detailed technical information needed for troubleshooting should errors occur.

The header includes useful information about the size of the image (`NAXIS1` & `NAXIS2`), the date (`DATE-OBS`), the exposure time (`EXPTIME`), where the celestial coordinates of where telescope was pointing (`RA` & `DEC`), and the optical filter in the light path (`FILTNAM`). The header is comprised of "keywords," e.g., `FILTNAM` with values on the right hand side of the equals sign.

The image data and the keywords are read into Python using the function `getdata()`, for example to read the file `d159.fits` into the variable `x` type

```
In [1]: import astropy.io.fits as pf
In [2]: fits1='d159.fits'
In [3]: x = pf.getdata(fits1)
In [4]: hdr = pf.getheader(fits1)
In [5]:
Out[5]: (1024, 1056)
In [6]: x.mean()
Out[6]: 1386.4204545454545
In [7]: x.min()
Out[7]: 1165.0
In [8]: x.max()
Out[8]: 65535.0
```

Note that even though the CCD has 2048 × 2048 pixels, the image is half this size because pixels on the CCD have been "binned," i.e., each 2 × 2 group of pixels in the original array is represented in the `FITS` file by a single value equal to the sum of these four.

Scrolling through the ASCII FITS header in a terminal is fine for browsing, but if you want to use information in the FITS header in your Python program, you will need to read the keywords. This is accomplished by using the function `open`, e.g., to find the object name

```
In [1]: import astropy.io.fits as pf
In [2]: fits1='d159.fits'
In [3]: hdulist = pf.open(fits1)
In [4]: hdulist.info()
Filename: d159.fits
No.    Name         Type      Cards   Dimensions   Format
0    PRIMARY     PrimaryHDU      93   (1056, 1024)   int16

In [5]: hdulist[0].header['object']
Out[5]: '109_954'

In [6]: hdulist[0].data
Out[6]:
array([[ 1228.,  1231.,  1227., …,  1208.,  1215.,  1215.],
       [ 1219.,  1225.,  1218., …,  1206.,  1210.,  1213.],
       [ 1224.,  1225.,  1209., …,  1220.,  1210.,  1214.],
       …,
       [ 1225.,  1221.,  1215., …,  1208.,  1214.,  1214.],
       [ 1233.,  1219.,  1218., …,  1212.,  1212.,  1220.],
       [ 1231.,  1235.,  1226., …,  1218.,  1204.,  1212.]], dtype=float32)


In [7]: hdulist.close()
```

The function `open` gets both the data and the header information. Notice the `close` operation when we are done with the file.

3.2 CCD binning and overscan

The Loral CCD camera has 2048 × 2048 pixels (see Table 1.) As we noted in the previous example, the CCD can be operated in a binning mode where blocks of adjacent pixels are combined to yield "superpixels." Binning is useful because it reduces the size of the resultant FITS file and also because it reduces the effects of read out noise. You can establish whether or not binning is enabled by examing the `CDELT1U` and `CDELT2U` keywords. For 2×2 binning

```
CDELT1U =                   -2 / COLUMN CHANGE PER PIXEL
CDELT2U =                   -2 / ROW CHANGE PER PIXEL
```

the resultant image should comprise 1024 × 1024 pixels.

A second factor that determines the size of the array in the FITS file is whether or not the CCD has been overscanned. This is recorded by the value of the `COVER` keyword. You may have noticed that the actual size of the array returned in the above example is 1024 × 1056 pixels, not 1024 × 1024—the are 32 additional columns in the FITS files that do not correspond to physical CCD pixels. This overscan region provides a way to correct the bias in the image. Oversan is implemented by continuing to clock the CCD row and column shift registers and digitizing the resultant signal even though all the pixels have been readout. No physical pixels to contribute charge to these dummy reads, so the resultant data values in the overscan region measure only the bias.

3.3 Bias, darks, & flats

At the beginning of each night various calibration images (or frames), including bias, darks, and flats are acquired. Bias frames are zero second exposures that provide a measure of the digital offset when the signal is zero. For dark exposures the shutter does not open; these images can be used to measure the dark current. Modern, science-grade CCDs typically have dark current of 0.01 e^- per second, or less, and for our purposes this systematic error is negligible. Flats are twilight sky images—both evening and morning twilight flats are available

One reason that the dark current is low is that the 40-inch CCD is cooled by liquid nitrogen (which boils at 77K at atmospheric pressure.) Unless someone forgot to fill the vacuum dewar that houses the CCD, it should be at its nominal operating temperature of about 160 K (-113°C). It is good practice to check the keyword:

```
TEMPDET =          -111.599998 / EXPOSURE START DETECTOR TEMPERATURE
```

and confirm that the CCD is cold.

The twilight sky provides a convenient source of spatially uniform illumination that can be used to measure the relative pixel-to-pixel response of the camera. Flats are constructed from sequences of short exposures (e.g., 5–10 s). Because the brightness of the twilight sky changes rapidly as the sun sets, care is needed get sufficient counts for good signal-to-noise, while avoiding saturation. Typically, telescope tracking is turned off for flats, so stars appear to move from one image to the next.

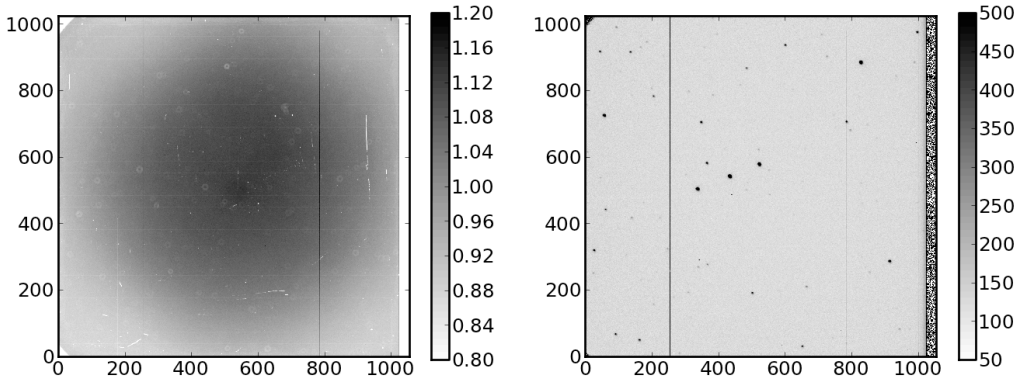

Figure 2: ***Left:*** **The *V*-band flat field image. The bias has been subtracted and the flat normalized so that the median value in the image is 1.0.** ***Right:*** **The bias-subtracted, flat-fielded image of a 20-s *V*-band exposure. The central star at pixel (440, 530) in the triplet of stars near the center of the image is the photometric standard star Landolt 109 954.**

4 A Star Field

In this example we choose a 20 s *V*-band image of the field containing the standard star Landolt 109-954[4] (RA 17h44m15.84s; DEC -00°02'16" J2000). The python script loads the image, subtracts the bias region, and divides by the appropriate flat (the flat is different for the *B*, *V*, *R*, and *I* filters):

[4] http://www.noao.edu/wiyn/obsprog/images/charts/c113.html

```
import astropy.io.fits as pf
import numpy as np
import matplotlib.pyplot as plt
from bsub import bsub
# File names #
fits1='d159.fits'
flat1='d121.fits'

x = pf.getdata(fits1)
hdr = pf.getheader(fits1)
xb = bsub(x,hdr.get('cover')) # Bias subtract

# Flat #
flat  = pf.getdata(flat1)
fhdr = pf.getheader(flat1)
flatb =  bsub(flat,hdr.get('cover')) # Bias subtract
flatb = flatb/np.median(flatb)   # normalize

plt.figure()
plt.subplot(121)
imp = plt.imshow(flatb,cmap='gray_r',vmin=0.8,vmax=1.2)
plt.colorbar()
plt.subplot(122)
imp = plt.imshow(xb/flatb,cmap='gray_r',vmin=50,vmax=500)
plt.colorbar()
```

I have not given the details of the bias subtraction function `bsub.py`, which uses the data in over-scan columns. Inspection of this image (Figure 2) reveals a few dozen stars. Compare this field with the region with the Aladin sky atlas[5] (Figure 3.)

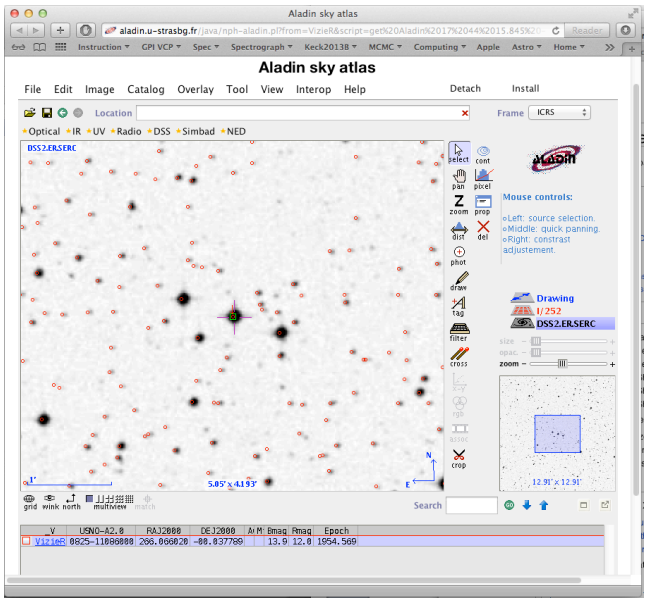

Figure 3: An image of the field of Landolt 109 954 from the Aladin sky atlas. A 1 arc minute scale bar is shown at the lower left. The same triplet of stars seen in Figure 2 is visible here at the center of the image. Note that normal astronomical convention is adopted here: north is up and east is to the left. The image in Figure 2 should be flipped in the *y*-direction to conform to this convention.

Comparison of Figure 2 and Figure 3 shows that the images from the Nickel telescope have to be flipped up/down using the [::-1] syntax so that they conform to the conventional orientation with

[5] http://aladin.u-strasbg.fr/

north at the top and east to the left. Also, to get `imshow` to display the images to display with pixel (0,0) at the bottom left hand corner you need to edit your `matplotlibrc` file[6]:

```
### IMAGES
#image.aspect : equal             # equal | auto | a number
#image.interpolation  : bilinear  # see help(imshow) for options
image.interpolation  : nearest  # see help(imshow) for options
#image.cmap   : jet               # gray | jet etc...
image.cmap   : gray              # gray | jet etc...
#image.lut    : 256               # the size of the colormap lookup table
image.origin : lower             # lower | upper
#image.resample  : False
```

4.1 Star positions

Once we have confirmed that the telescope was pointed in the right direction and the image corresponds approximately to the correct field, the next step is to identify and measure the *x*- and *y*-positions of the stars in the image. The location of stars are best measured using their centroids:

$$\langle x \rangle = \sum_i x_i I_i \Big/ \sum_i I_i \,, \quad \langle y \rangle = \sum_i y_i I_i \Big/ \sum_i I_i \,, \tag{1}$$

where the summation, i, runs only over a region in the vicinity of the star. Note that the denominator, ΣI_i, is a measure of the brightness of the star—this is useful information that should be saved along with the centroids.

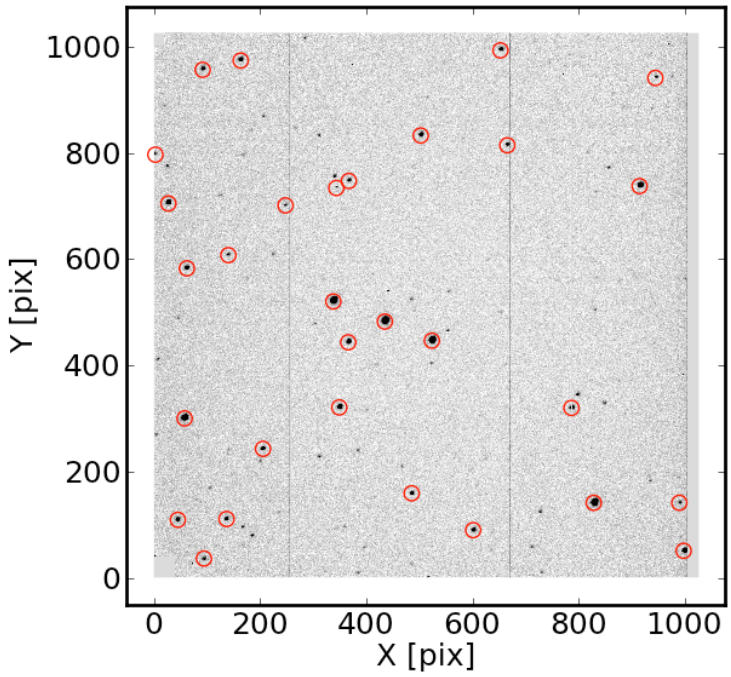

Figure 4: The *V*-band image of the field of Landolt 109 954 using a reverse (negative) color table `cmap = plt.cm.gray_r`. The red circles show the positions of 30 stars found by locating the brightest pixel, computing the center of light according to Eq. (1) and then moving on to the next star.

[6] http://matplotlib.org/users/customizing.html

4.2 Star catalogs

We can now compare the observed *x*- and *y*-positions (on our CCD image with the predicted positions of stars taken from a catalog, such as the US Naval Observatory USNO-B1.0 Catalog[7]. The USNO-B1.0 catalog lists positions, proper motions, and magnitudes in various optical pass-bands for 1,042,618,261 objects over the entire sky. The data are from scans of photographic plates from the Palomar 48-inch Schmidt telescope taken for various sky surveys during the last 50 years. The USNO-B1.0 catalog, which provides all-sky coverage down about $V = 21$ mag., and positional accuracy of about 0.2 arc second in the J2000 celestial coordinate system should be adequate for our purposes.

In this example the *x*- and *y*-centroids of stars are the dependent variable and their cataloged positions are the independent variable. By finding the relation between independent and dependent variable we will be able to measure the celestial coordinates for any location in our image.

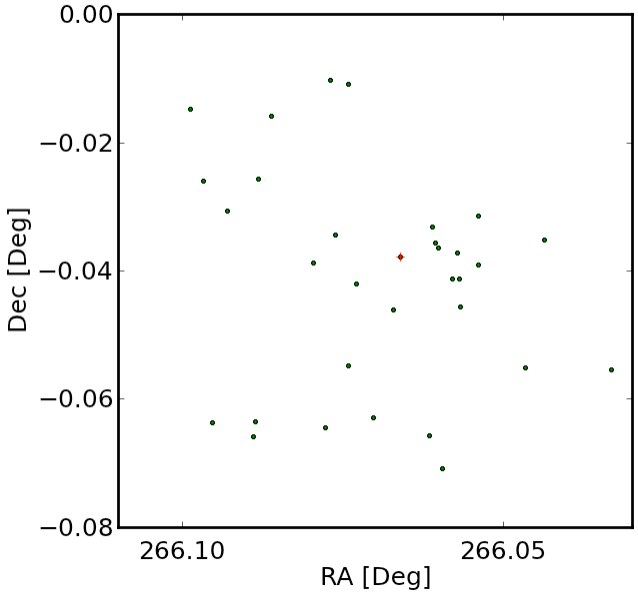

Figure 5: The positions of stars (green dots) in a 4 arc minute × 4 arc minute field selected from the USNO-B1.0 catalog (*r* < 17.0 mag.) The red + symbol shows the position of Landolt 109 954.

You can interrogate the USNO-B1.0 catalog from within Python (see § 8) using the web-based Vizier[8] astronomical catalog database as follows:

```
import astropy.io.fits as pf
import matplotlib.pyplot as plt
import numpy as np
import urllib as url

fits1='d159.fits'
s1 = pf.open(fits1)
```

[7] http://tdc-www.harvard.edu/software/catalogs/ub1.html
[8] http://vizier.hia.nrc.ca/viz-bin/VizieR

```
# Read position from the FITS file and convert RA/DEC to degrees
# be sure to check that the header data is reliable. If not
# edit the position by hand.
ras = s1[0].header['ra']
des = s1[0].header['dec']
radeg = 15*(float(ras[0:2]) + float(ras[3:5])/60. + float(ras[6:])/3600.)
dsgn = np.sign(float(des[0:3]))
dedeg = float(des[0:3]) + dsgn*float(des[4:6])/60. + dsgn*float(des[7:])/3600.

fovam = 3.0  # size of square search field in arc min
name,rad,ded,rmag =  usno(radeg,dedeg,fovam)

w = np.where(rmag < 17.)[0]   # select only bright stars r < 15 mag.

plt.plot(rad[w],ded[w],'g.')
plt.locator_params(axis='x',nbins=4)
plt.locator_params(axis='y',nbins=4)
plt.tick_params('x',pad=10)
plt.xlabel('RA [Deg]')
plt.ylabel('Dec [Deg]')
plt.ticklabel_format(useOffset=False)
plt.axis('scaled')
plt.xlim([266.11,266.03])   # reverse the x-axis direction
```

Note that by convention right ascension (*y* axis) increases to the left. The resultant plot for this example is shown in Figure 5. Inspection of this figure does not immediately give the impression that we are looking at the same star field that we see in Figure 3 and Figure 4. Look carefully at Figure 5 and convince yourself that you can trace the same pattern of stars. To make sure you have the right stars you can interrogate the USNO B-1 catalog directly from ALADIN, using the open/surveys menu choice. Clicking on a star in the ALADIN window will show you the catalog entry.

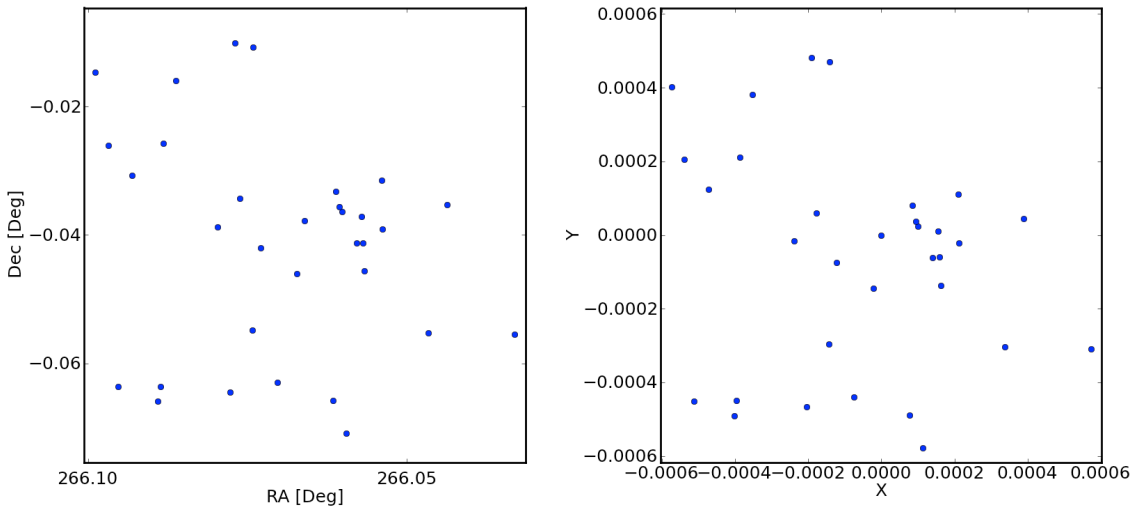

Figure 6: *Left*: The catalog positions of 32 USNO B-1 stars within a 4 × 4 arc minute square field centered at the position of Landolt 109 954 with $r < 17$ mag. *Right:* the same data plotted in standard coordinates, which is the projection from celestial coordinates onto the unit sphere tangent plane (see §4.3 Eq. (2)).

4.3 Matching stars & standard coordinates

We now have the positions of stars on the CCD in pixel *x* and *y* and in right ascension and declination from the USNO B-1 catalog. The USNO B-1 coordinates are on the celestial sphere, whereas the stars in our CCD image are measured on a plane, which is a projection of the

celestial sphere. The first step in the comparison is to project the USNO B-1 coordinates from the celestial sphere to the plane represented by the CCD so that we can make a one-to-one comparison of these quantities.

The transformation can be obtained by considering the geometry in Figure 7 (see the detailed derivation in §6). The projected coordinates (X, Y) of a position on the celestial sphere (α, δ) in the vicinity of a field centered at (α_0, δ_0) are

$$X = -\frac{\cos\delta \sin(\alpha - \alpha_0)}{\cos\delta_0 \cos\delta \cos(\alpha - \alpha_0) + \sin\delta \sin\delta_0}$$
$$Y = -\frac{\sin\delta_0 \cos\delta \cos(\alpha - \alpha_0) - \cos\delta_0 \sin\delta}{\cos\delta_0 \cos\delta \cos(\alpha - \alpha_0) + \sin\delta \sin\delta_0}, \quad (2)$$

where the celestial sphere is taken to be the unit radius.

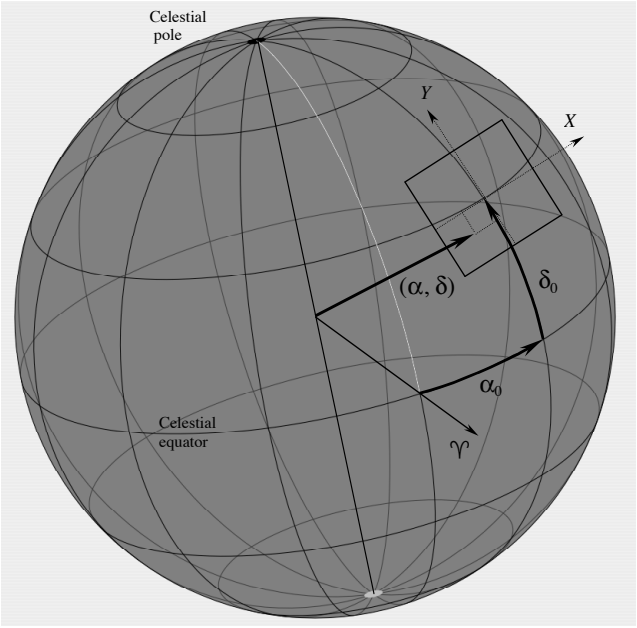

Figure 7: Projection from celestial coordinates of a point (α, δ) on the sky in a field centered at (α_0, δ_0) onto the tangent plane of a CCD with a position (X, Y). The X-axis is aligned along the small declination circle and the Y-axis is aligned along the great hour circle, and points towards the celestial pole. The symbol γ indicates the direction of the vernal equinox, which defines the origin of right ascension.

Conversion from (X, Y) coordinates to pixel coordinates (x, y), is straightforward. If f is the focal length of the camera and p is the pixel size, then for an ***ideal*** camera

$$x = f\left(X/p\right) + x_0$$
$$y = f\left(Y/p\right) + y_0 \quad ,$$

where, by ideal we mean that the focal length of the imaging system is constant over the field, there is no anamorphic magnification ($f = f_x = f_y$), or equivalently the pixels are square ($p_x = p_y$), and that the CCD is oriented so that the X-axis lies along the declination small circle and the Y-axis points north. In general, none of these conditions are true. Optical systems have variable magnification and distortion, pixels may be parallelograms, and the CCD is subject to an unknown rotation with respect to celestial coordinates. However, for a first guess let's adopt the nominal values from Table 1: f = 16,840 mm and p = 0.015 mm. The results of this comparison are shown in Figure 8.

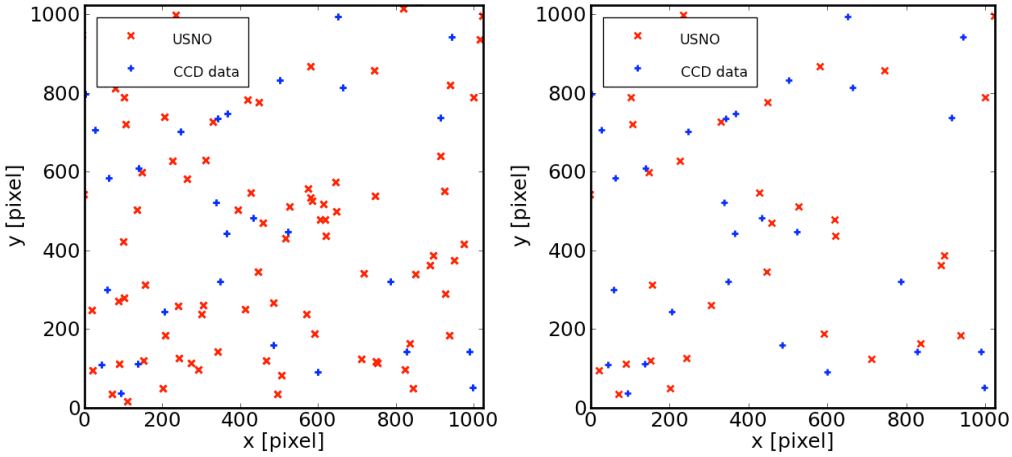

Figure 8: ***Left:*** **Comparison of the measured pixel positions (+) of stars and converted standard positions from the USNO B-1 catalog (x) using f = 16,840 mm, p = 0.015 mm (x2 binning), and $x_0 = y_0 = 512$. The situation is obviously complicated, but many of the CCD stars appear to have counterparts in the USNO B-1 catalog.** ***Right:*** **Plotting only the brighter stars (r < 16 mag.) reveals a clearer picture.**

Evidently, if we plot all the stars (left) then it is difficult to figure out how the star patterns match up; however, if we choose only some of the brighter stars from the USNO B-1 list (r < 16 mag.) it is clear that that we see the same pattern with an offset. Inspection of the stars also suggests that the magnification is not quite right and that two patterns are rotated with respect to one another. By using the distance between the predicted and measured positions it is possible to uniquely match up the stars (Figure 9).

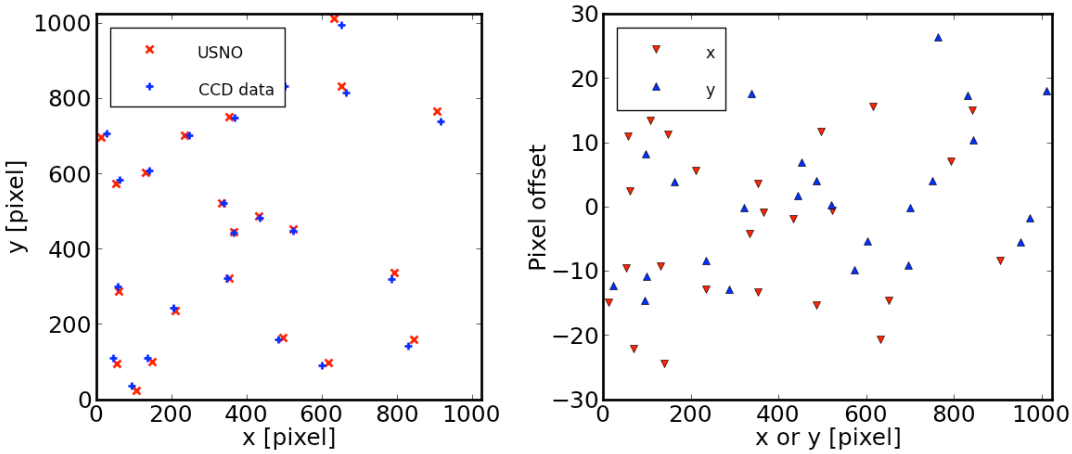

Figure 9: ***Left:*** **Stars from from Figure 8 matched using a distance criterion.** ***Right:*** **The pixel offset between the observed and predicted postitions of USNO B-1 stars versus or *x* or *y* pixel. The left panel shows systematic departures between the predicted and observed positions, which appears primarilly as a rotation.**

5 Least squares

The standard coordinates are defined for the tangent plane to the unit sphere. As we have seen in § 4.3 the positions on the CCD depend on the focal length and the pixel size. In general the CCD is not perfectly aligned so there is a rotation between the (x, y) CCD-based coordinate system and the (X, Y) standard coordinates. Taking this into account

$$x = \frac{f}{p}\left(X\cos\theta - Y\sin\theta\right) + x_0$$
$$y = \frac{f}{p}\left(X\sin\theta + Y\cos\theta\right) + y_0 \quad ,$$

where θ is the rotation between the two frames. The general affine transformation between two vector spaces, **X**, and **x** is described by a linear transformation (magnification, shear, rotation) plus a translation (see §7). This transformation is conveniently implemented by matrix multiplication using homogeneous coordinates, where $\mathbf{X} = (X, Y, 1)$, $\mathbf{x} = (x, y, 1)$ and

$$\mathbf{x} = \mathbf{T}\mathbf{X} \tag{3}$$

where

$$\mathbf{T} = \begin{pmatrix} (f/p)a_{11} & (f/p)a_{12} & x_0 \\ (f/p)a_{21} & (f/p)a_{22} & y_0 \\ 0 & 0 & 1 \end{pmatrix}. \tag{4}$$

Thus, in the linear approximation there are six unknown "plate constants", which we can solve for using the method of linear least squares. The constants a_{ij} refer to the scale, shear and orientation of the image, and x_0 and y_0 are pointing offsets in pixels.

To find the least-squares solution we write down the equations of condition describing the relation between the independent variables (X_i, Y_i) and our N position measurements of the dependent variable from the CCD image, (x_i,y_i), as

$$\begin{aligned} x_1 &= (f/p)a_{11}X_1 + (f/p)a_{12}Y_1 + x_0 \\ x_2 &= (f/p)a_{11}X_2 + (f/p)a_{12}Y_2 + x_0 \\ \cdots &= \cdots \\ x_N &= (f/p)a_{11}X_N + (f/p)a_{12}Y_N + x_0, \end{aligned} \tag{5}$$

and

$$\begin{aligned} y_1 &= (f/p)a_{21}X_1 + (f/p)a_{22}Y_1 + y_0 \\ y_2 &= (f/p)a_{21}X_2 + (f/p)a_{22}Y_2 + y_0 \\ \cdots &= \cdots \\ y_N &= (f/p)a_{21}X_N + (f/p)a_{22}Y_N + y_0. \end{aligned} \tag{6}$$

Or in compact matrix form for Eq. (5) and Eq. (6) for can be represented as

$$\mathbf{a} = \mathbf{Bc}, \tag{7}$$

where for x we have

$$\mathbf{a} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{pmatrix}, \quad \mathbf{B} = \begin{pmatrix} (f/p)X_1 & (f/p)Y_1 & 1 \\ (f/p)X_2 & (f/p)Y_2 & 1 \\ \vdots & \vdots & \vdots \\ (f/p)X_N & (f/p)Y_N & 1 \end{pmatrix}, \text{ and } \mathbf{c} = \begin{pmatrix} a_{11} & a_{12} & x_0 \end{pmatrix},$$

with a similar equation for y.

It is easy to compute χ^2 in matrix form using Eq. (7)

$$\chi^2 = (\mathbf{a} - \mathbf{Bc})^T (\mathbf{a} - \mathbf{Bc}) = \left\| \mathbf{a} - \mathbf{Bc} \right\|_2^2, \tag{8}$$

Where the $\|\cdot\|_2$ notation represents the vector norm or length from Pythagoras' theorem. Using the matrix derivative result[9]

[9] http://matrixcookbook.com: The Matrix Cookbook, K. B. Petersen & M. S. Pedersen, V. 2012/11/15

$$\frac{\partial}{\partial \mathbf{X}}\left|\left|\mathbf{X}-\mathbf{Y}\right|\right|_2 = \frac{\mathbf{X}-\mathbf{Y}}{\left|\left|\mathbf{X}-\mathbf{Y}\right|\right|_2}\ , \tag{9}$$

we can find the condition that minimizes χ^2

$$\frac{\partial}{\partial \mathbf{c}}\left|\left|\mathbf{a}-\mathbf{B}\mathbf{c}\right|\right|_2 = \frac{\mathbf{a}-\mathbf{B}\mathbf{c}}{\left|\left|\mathbf{a}-\mathbf{B}\mathbf{c}\right|\right|_2} = 0 \tag{10}$$

The matrix **B** is not square, so no inverse exists. However, we can solve for **c** from by multiplying each side Eq. (10) by the transpose of **B**, $\mathbf{B}^T$, followed by the inverse matrix of the resulting square matrix $(\mathbf{B}^T\mathbf{B})^{-1}$, i.e.,

$$\mathbf{c} = \left(\mathbf{B}^T\mathbf{B}\right)^{-1}\mathbf{B}^T\mathbf{a}\ . \tag{11}$$

A similar expression is used for *y*.

The results of applying Eq. (11) to find the matrix elements of **c** are summarized in Figure 10. In the initial match most stars show very small residuals, suggesting that the match is reliable except perhaps for three stars that may be mismatched. When these are removed the overall quality of the fit is significantly improved.

The coefficients of the solution are shown in Table 2, assuming the nominal values *f/p* =561,000 pixels per radian. Recall that the geometric interpretation of the determinant of a 2-d transformation is the scale factor for area (see Eq. (23)). Therefore, a convenient measure of the effective telescope scale *f/p* (in pixels per radians) is

$$f/p = \sqrt{\det(\mathbf{T})}\ , \tag{12}$$

where **T** is the matrix in Eq. (3) and the matrix elements of **T** are found using Eq. (11). Assuming that the pixel size is known with precision then the effective focal length is about 1% shorter than the nominal value. Inspection of the matrix elements in Table 2 shows that **T** is very nearly skew symmetric, which means that pixels are square to better than a part in a thousand. The rotation of CCD columns from true north is about 2.3 degree counterclockwise (see Eq. (21)).

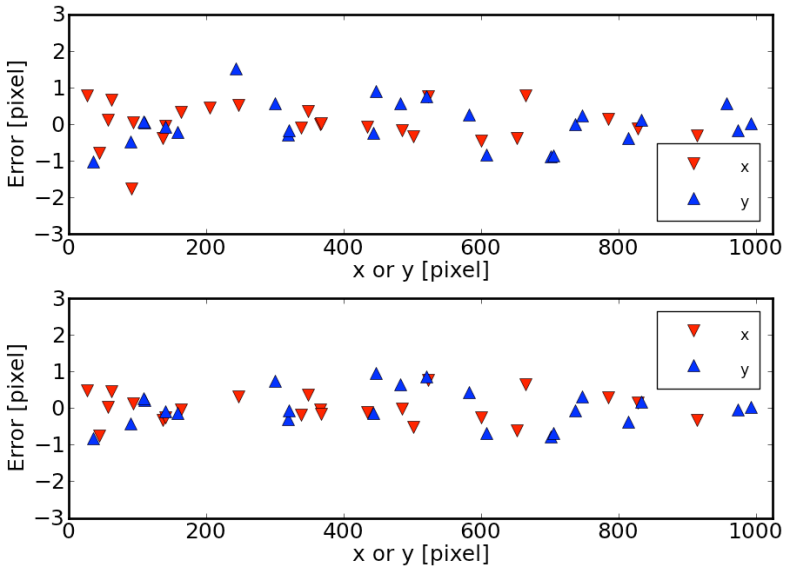

Figure 10: ***Top:*** **Residual errors in *x* and *y* relative to a linear least squares solution. In the top plot the all 25 stars are matched between the CCD and the USNO B-1 catalog. Of these matches two have suspiciously large residuals (>1.5 pixels). The stars with poor matches may be false matches.** ***Bottom:*** **If we remove matches with residuals larger than 1.5 pixels then the bottom plot results. . The rms *x* and *y* errors are 0.39 pixels and 0.50 pixels respectively.**

The least-squares solution allows us to compute the standard coordinates for any pixel in the CCD. To convert from a measured (x, y) CCD position to (α, δ) in celestial coordinate we first convert to standard coordinates, using the inverse of Eq. (3), i.e.,

$$\mathbf{X} = \mathbf{T}^{-1}\mathbf{x}\,. \tag{13}$$

The conversion from standard coordinates to celestial coordinates is then given by Eq. (18) (see §6).

Table 2: Least squares coefficients.

	x	*y*
	$a_{11} = 0.99089$ $a_{12} = 0.00399$ $x_0 = 435.02$	$a_{21} = -0.00398$ $a_{22} = 0.98940$ $y_0 = 482.42$
rms error	0.54 pix	0.59 pix
$\sqrt{\det(\mathbf{T})}$	0.9908	

6 Appendix: Tangent plane projection

Consider the Cartesian coordinate system (x, y, z) with the x-axis pointed towards some right ascension α_0, and the x-y plane coincident with the celestial equator. The geometry is illustrated in Figure 11.

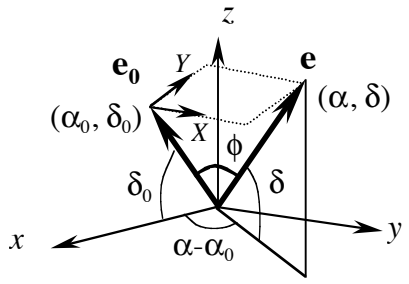

Figure 11: A coordinate system with the x-axis pointed towards some right ascension α_0, and the x-y plane coincident with the celestial equator. The angle between the directions (α, δ) and (α_0, δ_0) is ϕ. The tangent plane is normal to the unit vector $\mathbf{e}_0$. The tangent plane coordinates are (X, Y).

The components of the unit vector **e** pointing towards a star at (α, δ) are

$$\begin{aligned} x &= \cos\delta\cos(\alpha-\alpha_0) \\ y &= \cos\delta\sin(\alpha-\alpha_0)\ . \\ z &= \sin\delta \end{aligned} \tag{14}$$

The angle between the directions **e**(α, δ) and $\mathbf{e}_0(\alpha_0, \delta_0)$ is ϕ is given by computing the dot product

$$\begin{aligned} \cos\phi &= \mathbf{e}_0\cdot\mathbf{e} \\ &= \cos\delta\cos(\alpha-\alpha_0)\cos\delta_0+\sin\delta_0\sin\delta. \end{aligned} \tag{15}$$

Consider the plane normal to $\mathbf{e}_0$ with X oriented along the y-axis and Y rotated about y by an angle equal to δ_0 such that the unit vectors in this plane are

$$\hat{\mathbf{X}} = \begin{pmatrix} 0 & 1 & 0 \end{pmatrix};\quad \hat{\mathbf{Y}} = \begin{pmatrix} \sin\delta_0 & 0 & -\cos\delta_0 \end{pmatrix}.$$

A point in the tangent plane **p** is

$$\mathbf{p} = \mathbf{e}_0 - \left(X\hat{\mathbf{X}} + Y\hat{\mathbf{Y}}\right) = p\mathbf{e}\ ,$$

where $p = 1/\cos\phi = (1+X^2+Y^2)^{1/2}$.

When written in component form we have three simultaneous equations relating (α, δ) and (X, Y)

$$\begin{pmatrix} \cos\delta_0 - Y\sin\delta_0 \\ -X \\ \sin\delta_0 - Y\cos\delta_0 \end{pmatrix} = \begin{pmatrix} \dfrac{\cos(\alpha-\alpha_0)\cos\delta}{\cos(\alpha-\alpha_0)\cos\delta\cos\delta_0 + \sin\delta\sin\delta_0} \\ \dfrac{\sin(\alpha-\alpha_0)\cos\delta}{\cos(\alpha-\alpha_0)\cos\delta\cos\delta_0 + \sin\delta\sin\delta_0} \\ \dfrac{\sin\delta}{\cos(\alpha-\alpha_0)\cos\delta\cos\delta_0 + \sin\delta\sin\delta_0} \end{pmatrix}, \tag{16}$$

which yield

$$\begin{aligned} X &= -\frac{\sin(\alpha-\alpha_0)\cos\delta}{\cos(\alpha-\alpha_0)\cos\delta\cos\delta_0 + \sin\delta\sin\delta_0} \\ Y &= -\frac{\cos(\alpha-\alpha_0)\cos\delta\sin\delta_0 - \sin\delta\cos\delta_0}{\cos(\alpha-\alpha_0)\cos\delta\cos\delta_0 + \sin\delta\sin\delta_0} \end{aligned}. \tag{17}$$

Expressions for α and δ as a function of X and Y can be found from

$$\begin{aligned} \tan(\alpha-\alpha_0) &= -\frac{X}{\cos\delta_0 - Y\sin\delta_0} \\ \sin\delta &= \frac{\sin\delta_0 + Y\cos\delta_0}{\left(1+X^2+Y^2\right)^{1/2}} \end{aligned}, \tag{18}$$

by using inverse trig functions

7 Appendix: Affine transformations

The affine transformations described by Eq. (3) comprise a translation

$$\mathbf{T} = \begin{pmatrix} 1 & 0 & \Delta x \\ 0 & 1 & \Delta y \\ 0 & 0 & 1 \end{pmatrix} \tag{19}$$

by $(\Delta x, \Delta y)$, a magnification

$$\mathbf{M} = \begin{pmatrix} m_x & 0 & 0 \\ 0 & m_y & 0 \\ 0 & 0 & 1 \end{pmatrix}, \tag{20}$$

where m_x and m_y are the magnification in the x- and y-directions, a rotation,

$$\mathbf{R} = \begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix}, \tag{21}$$

where θ is a counterclockwise rotation, and a shear ,

$$\mathbf{S}_h = \begin{pmatrix} 1 & s_h & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \text{ and } \mathbf{S}_v = \begin{pmatrix} 1 & 0 & 0 \\ s_v & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \tag{22}$$

where s_h is the horizontal shear (x-displacement proportional to y-position.) and s_v is the vertical shear. The product of these matrices (magnification, rotation, shear, & translation) yield:

$$\mathbf{TS_hS_vRM} = \begin{pmatrix} m_x\left[\left(1 + s_h s_v\right)\cos\theta + s_h\sin\theta\right] & m_y\left[s_h\cos\theta - \left(1 + s_h s_v\right)\sin\theta\right] & \Delta x \\ m_x\left(s_v\cos\theta + \sin\theta\right) & m_y\left(\cos\theta\text{-}s_v\sin\theta\right) & \Delta y \\ 0 & 0 & 1 \end{pmatrix}$$

and the determinant is

$$\det\left(\mathbf{TS_hS_vRM}\right) = m_x m_y . \tag{23}$$

Thus, the determinant represents the scale factor by which areas are transformed.

8 Querying the USNO-B1.0 catalog

The following Python program will return the star names, the celestial coordinates (RA and DEC in decimal degrees; equinox J2000.0) and the *r*-band magntiude given a search location (also decimal degrees; J2000.0) and the search field (a square region in arc minutes). This routine queries the web-based Vizier astronomical catalog system.

```
def usno(radeg,decdeg,fovam,epoch):

    # James R. Graham 2013/10/13 UC Berkeley
    # get USNO B-1 stars centered at radeg and decdeg (J2000.0) in degrees centered
    # in a square field of view (arc min). Corrects for proper motion to current epoch.

    import urllib as url
    import numpy as np
    import string as str
    import pdb

    str1 = 'http://webviz.u-strasbg.fr/viz-bin/asu-tsv/?-source=USNO-B1'
    str2 = '&-c.ra={:4.6f}&-c.dec={:4.6f}&-c.bm={:4.7f}/{:4.7f}&-
out.max=unlimited'.format(radeg,decdeg,fovam,fovam)

    str = str1+str2
    print 'Calling Vizier',str
    f = url.urlopen(str)

#   Read from the object, storing the page's contents in 's'.
    s = f.read()
    f.close()

    sl = s.splitlines()
    sl = sl[45:-1]       # get rid of header  - updated Oct 2013
    name = np.array([])
    rad  = np.array([])  # RA in degrees
    ded  = np.array([])  # DEC in degrees
    rmag = np.array([])  # rmage

    for k in sl:
        kw = k.split('\t')
        ded0 = float(kw[2])
        pmrad = float(kw[6])/3600e3/np.cos(np.deg2rad(ded0))   # comvert from mas/yr to deg/year
        pmded = float(kw[7])/3600e3

        name  = np.append(name,kw[0])
        rad   = np.append(rad,float(kw[1]) + pmrad*(epoch-2000.0)     )
        ded   = np.append(ded,float(kw[2]) + pmded*(epoch-2000.0)     )

        if kw[12] != '     ':             # case when no mag is reported
            rmag = np.append(rmag,float(kw[12]))
        else:
            rmag = np.append(rmag,np.nan)

    return name,rad,ded,rmag
```

9 Appendix: Web Resources

Find what small bodies are observable at: http://ssd.jpl.nasa.gov/sbwobs.cgi

US Naval Observatory (home of the USNO B astrometric catalog): http://ad.usno.navy.mil/

ALADIN catalog and sky viewer: http://aladin.u-strasbg.fr/aladin.gml

10 On Line Monographs

Astronomy on the Personal Computer, Montenbruck & Pfleger, Springer (partial preview at http://books.google.com/books?id=WDjJIww337EC)

Modern Astrometry, Jean Kovalevsky, Springer (partial preview at http://books.google.com/books?id=s4azHlUeIYgC)